ESTRUCTURA DE DATOS 1
Código ST0245

Taller en Sala 6
Listas hechas con arreglos

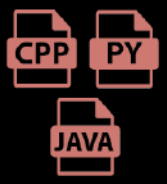
Objetivo: **1.** Solucionar problemas del mundo real con algoritmos. **2.** Escoger la estructura de datos adecuada para resolver un problema dado. **3.** Usar la notación O para encontrar formalmente la complejidad asintótica en tiempo de algoritmos.

Consideraciones: Lean y verifiquen las consideraciones de entrega,

Trabajo en Parejas

Mañana, plazo de entrega

Docente entrega plantilla de código en GitHub



Sí .cpp, .py o .java

No .zip, .txt, html o .doc

Alumnos entregan código sin comprimir GitHub

En la carpeta Github del curso, hay un código iniciado y un código de pruebas (tests) que pueden explorar para solucionar los ejercicios

Estructura del documento: **a)** Datos de *vida real*, **b)** I*ntroducción* a un problema, **c)** Problema a resolver, **d)** Ayudas. Identifiquen esos elementos así:

a)

b)

c)

d)

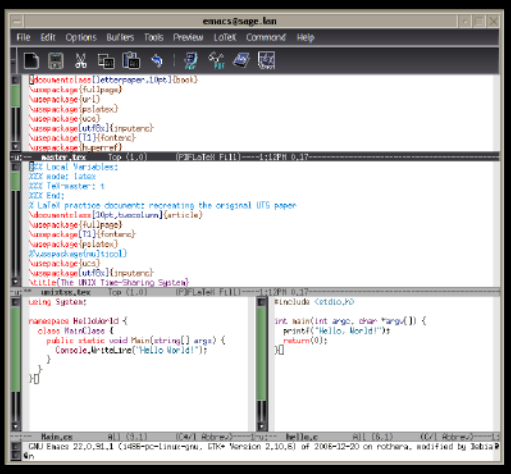
PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

Ejercicios a resolver

En la vida real, se utilizan como estructura de datos como *gap buffer* para representar el texto en el editor *Emacs*. Otra estructura de datos que se utiliza en procesadores de palabras como *Microsoft Word* y *Google Docs* se llama *rope* o *cord*. Ver https://bit.ly/2tZ2Lqc

1 Para representar un programa en un editor de texto como *Emacs* es necesario utilizar una estructura dinámica porque el tamaño del documento crece conforme el usuario escribe nuevos caracteres o líneas.

Adicionalmente, el usuario debe tener la posibilidad de eliminar un caracter, causando que se mueva todo lo que sigue de él hacia la atrás. El usuario también debe poder insertar un carácter, causando que todo que sigue de él se mueva hacia adelante.



Implementen una lista implementada con arreglos para un editor de texto que permita insertar y borrar caracteres en cualquier posición. ¿La complejidad del método *agregar caracter* permite que su lista sea utilizada en un editor de texto? ¿Cuál es la complejidad de agregar n caracteres?

2 **[Ejercicio Opcional] Implementen un algoritmo que lea una cantidad indeterminada de enteros usando *Scanner* y los guarde en una lista implementada con arreglos de forma invertida, es decir, primero los últimos y luego los primeros. Al ingresar -1 debe terminar la lectura de los números.** ¿Qué complejidad tiene este algoritmo? Finalmente, impriman la lista implementada con arreglos para verificar que funciona correctamente el algoritmo.

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

[Ejercicio Opcional] **Implementen un algoritmo que reciba por parámetro un número N > 0 y genere una lista implementada con arreglos con el patrón {1, 1, 2, 1, 2, 3, … 1,2,3..n}.**

Finalmente, impriman la lista implementada con arreglos para verificar que funcional el algoritmo.

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

Ayudas para resolver los Ejercicios

Ejercicio 1

Pista 0: Utilicen este simulador de *ArrayList*: http://www.cs.armstrong.edu/liang/animation/web/ArrayList.html

Pista 1: Implementen su propia clase *ArrayList,* por simplicidad, sólo para datos enteros. En particular, un constructor, el método *size*, el método *get* y el método *add*.

Pista 2: Utilicen el método *Arrays.copyOf(…)*

Pista 3: Lancen la excepción *IndexOutOfBoundsException* de ser necesario

Pista 4: Consideren este blog que explica la implementación de un *ArrayList* en Java: https://netjs.blogspot.com/2015/08/how-arraylist-works-internally-in-java.html

Pista 5: Consideren este blog que explica la complejidad de las listas de Python: http://interactivepython.org/runestone/static/pythoned/AlgorithmAnalysis/Listas.html

Pista 6: Diferencien MiArrayList *de ArrayList* del API de Java. Un error común es creer que todo se soluciona llamando los métodos existentes en *ArrayList* y, no es así, la idea es implementar una lista con arreglos nosotros mismos. A continuación, un ejemplo del error:

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

```java
// Retorna el tamaño actual de la lista
public int size()
{
    return size();
}

// Retorna el elemento en la posición index
public int get(int index)
{
    return get(index);
}

// Inserta un dato en la posición index
public void insert(int data, int index)
{
    if (index <= size())
    {
        insert(data,index);
    }
}

// Borra el dato en la posición index
public void remove(int index)
{
    remove(index);
}

// Verifica si está un dato en la lista
public boolean contains(int data)
{
    return contains(data);
}
```

Ejemplo 1: Para el ArrayList [1,2,3,4], el método get(1) retorna el número 2, el método add(2,5) cambia el ArrayList a [1,2,5,3,4], el método add(6) cambia el ArrayList a [1,2,5,3,4,6] y el método size() retorna 6.

Ejercicio 3

Ejemplos:

N=1 [1]
N=2 [1, 1, 2]
N=4 [1, 1, 2, 1, 2, 3, 1, 2, 3, 4]

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

¿Alguna inquietud?

CONTACTO

Docente Mauricio Toro Bermúdez
Teléfono: (+57) (4) 261 95 00 Ext. 9473
Correo: mtorobe@eafit.edu.co
Oficina: 19- 627

Agenden una cita dando clic en la pestaña
-*Semana*- de http://bit.ly/2gzVg10